Deterministic ascending tree automata I

By J. VIRÁGH

1. Introduction

In the early 60s Automata Theory was considerably influenced by the methods and results of Universal Algebra. In fact, if we regard the input signs as unary operational symbols over the state set, then the automaton can be identified with a special universal algebra (unoid). Allowing non-unary input signs Thatcher and Wright [7] and Doner [3] came to the notion of the generalized or tree automata which accept arbitrary trees instead of the linear words of ordinary automata.

Two types of tree automata are investigated in the literature. The first one, the descending tree automata (known also as frontier-to-root or sinking automata, cf. [3], [6], [7]) proceed the input trees from the leaves along all branches towards the root. All results of the 'classical theory' such as the equivalence of the deterministic and nondeterministic devices, the minimization algorithm, Nerode's theorem, regular (tree) grammars, regular expressions and the Kleene-theorem can be generalized for this type of automata (cf. [1]—[3] and [6], [7]).

A less investigated generalization led to the notion of the ascending (called also root-to-frontier or climbing) tree automata, cf. [4], [6]. This device reads the input trees starting at the root proceeding then towards the leaves along the branches. Our investigations were inspired by the results of Magidor and Moran [6] especially by Section 6 of their paper.

Our aim is to generalize the results of the classical theory for ascending tree automata. In this part I we investigate a generalization of the Kleene-theorem and the characterization of sets accepted by ascending tree automata as sets generated by special regular tree grammars. The algebraic notations developed by Gécseg and Steinby in [4] will be used throughout this paper. Nullary operations will be excluded. This restriction is necessary by some investigations concerning ascending tree automata, cf. [4].

2. Preliminaries

For arbitrary set A, $\mathcal{P}(A)$ denotes the power set of A. N stands for the set of all positive integers, i.e., $N=\{1, 2, 3, \dots\}$.

Let F be a finite nonvoid set and r a mapping of F into N. We call the ordered pair $\langle F, r\rangle$ a *type*. The elements of F are the *operational symbols*. If $f\in F$ and

$r(f)=n$, then we say that the arity of f is n or f is an *n-ary operational symbol.* We often refer to the type $\langle F, r\rangle$ simply by F and we take the set F as the (disjoint) union $\bigcup (F_n | n\in N)$, where F_n is the set of all n-ary operational symbols from F.

Let $X=\{x_1, x_2, \ldots\}$ be a countable set of variables and $X_n=\{x_1, x_2, \ldots, x_n\}$ for every $n\in N$. We define the set $T_{F,n}$ of *n-ary F-trees* as the smallest set satisfying

(i) $X_n \subseteq T_{F,n}$ and

(ii) $f(p_1, \ldots, p_m)\in T_{F,n}$ whenever $p_1, \ldots, p_m\in T_{F,n}$ and $f\in F_m$ for some $m\in N$.

We note that the set $T_{F,n}$ is identical with the set of all n-ary polinomial symbols of type F in the sense of Grätzer [5]. The subsets of $T_{F,n}$ are called *n-ary F-forests* or simply forests when n and F are specified by the context. T_F stands for the set $\bigcup(T_{F,n} | n\in N)$.

Next we define devices capable of recognizing forests. To this we need some preparations.

Let F be an arbitrary type. The ordered pair $\mathfrak{A}=\langle A, F^{\mathfrak{A}}\rangle$ is called a *nondeterministic F-algebra* if A is a nonempty set and $F^{\mathfrak{A}}=(f^{\mathfrak{A}} | f\in F)$ is a set of nondeterministic operations on A, i.e., if $f\in F_k$, then $f^{\mathfrak{A}}$ is a mapping

$$f^{\mathfrak{A}}\colon A^k \to \mathcal{P}(A).$$

In that special case when $f(a_1, \ldots, a_k)$ is a singleton for every $f\in F$ and $(a_1, \ldots, a_k)\in A^k$, we speak about *$F$-algebra.* Identifying the singletons with their elements (and that will be our practice in the following discussions) we can define an F-algebra as a system $\mathfrak{A}=\langle A, F^{\mathfrak{A}}\rangle$, where every operation is a mapping

$$f^{\mathfrak{A}}\colon A^k \to A.$$

The triple $\mathbf{A}=\langle \mathfrak{A}, \mathbf{a}, A'\rangle$ is called an *n-ary nondeterministic descending F-automaton* if

(i) $\mathfrak{A}=\langle A, F^{\mathfrak{A}}\rangle$ is a nondeterministic F-algebra, whose carrier A is called the *state set* of the automaton,

(ii) $\mathbf{a}=(A^{(1)}, A^{(2)}, \ldots, A^{(n)})\in(\mathcal{P}(A))^n$ is the *initial vector,*

(iii) $A'\subseteq A$ is the *set of final states.*

If $\mathfrak{A}$ is an F-algebra and all components of the initial vector are singletons, then we say that $\mathbf{A}=\langle \mathfrak{A}, \mathbf{a}, A'\rangle$ is an *n-ary deterministic descending F-automaton.*

Every automaton $\mathbf{A}$ induces a mapping $\beta^{\mathbf{A}}\colon T_{F,n}\to\mathcal{P}(A)$ in the following manner:

(i) $\beta^{\mathbf{A}}(x_j) = A^{(j)} \quad (x_j\in X_n, \quad j = 1, 2, \ldots, n),$

(ii) $\beta^{\mathbf{A}}(p) = \bigcup(f^{\mathfrak{A}}(b_1, \ldots, b_k) | b_j\in\beta^{\mathbf{A}}(p_j), \quad j = 1, 2, \ldots, k)$ if $p = f(p_1, \ldots, p_k)$.

Then $T(\mathbf{A})=\{p | p\in T_{F,n} \,\&\, \beta^{\mathbf{A}}(p)\cap A'\neq\emptyset\}$ is the *forest recognized by* $\mathbf{A}$. If $\mathbf{A}$ is deterministic, then $\beta^{\mathbf{A}}(p)$ is a singleton for every $p\in T_{F,n}$ and $T(\mathbf{A})$ can be written as

$$T(\mathbf{A}) = \{p | p\in T_{F,n} \,\&\, \beta^{\mathbf{A}}(p)\in A'\}.$$

Now we present the necessary definitions for our second type of devices, the so-called ascending tree automata.

The ordered pair $\mathfrak{B}=\langle B, F^{\mathfrak{B}}\rangle$ is called a *nondeterministic ascending F-algebra* if B is a nonempty set and $F^{\mathfrak{B}}=(f^{\mathfrak{B}} | f\in F)$ is a set of nondeterministic ascending operations on B, i.e., if $f \in F_k$, then $f^{\mathfrak{B}}$ is a mapping

$$f^{\mathfrak{B}}\colon B \to \mathscr{P}(B^k).$$

Again, if for all $f \in F$ and $b \in B$, $f^{\mathfrak{B}}(b)$ is a singleton, then we say that $\mathfrak{B}$ is a *deterministic ascending F-algebra* and we write the operations as mappings

$$f^{\mathfrak{B}}\colon B \to B^k.$$

The triple $\mathbf{B} = \langle \mathfrak{B}, B', \mathbf{b} \rangle$ is called an *n-ary* nondeterministic ascending F-automaton, if

(i) $\mathfrak{B} = \langle B, F^{\mathfrak{B}} \rangle$ is a nondeterministic ascending F-algebra, and B is called the *state set* of $\mathbf{B}$,

(ii) $B' \subseteq B$ is the *set of initial states,*

(iii) $b = (B^{(1)}, B^{(2)}, \ldots, B^{(n)}) \in (\mathscr{P}(B))^n$ is the *vector of final states.*

If $\mathfrak{B}$ is a deterministic ascending F-algebra and B' is a singleton, then we say that $\mathbf{B} = \langle \mathfrak{B}, B', \mathbf{b} \rangle$ is an *n-ary deterministic ascending F-automaton.*

With every ascending automaton $\mathbf{B}$ we associate a mapping $\alpha^{\mathbf{B}}\colon T_{F,n} \to \mathscr{P}(B)$ as follows:

(i) $\alpha^{\mathbf{B}}(x_j) = B^{(j)}$ $(x_j \in X_n,\ \ j = 1, 2, \ldots, n)$,

(ii) $\alpha^{\mathbf{B}}(p) = \{b \mid f^{\mathfrak{B}}(b) \cap \alpha^{\mathbf{B}}(p_1) \times \alpha^{\mathbf{B}}(p_2) \times \ldots \times \alpha^{\mathbf{B}}(p_k) \neq \emptyset\}$

if $f \in F_k$ and $p = f(p_1, p_2, \ldots, p_k)$.

The forest recognized by $\mathbf{B}$ is defined by

$$T(\mathbf{B}) = \{p \mid p \in T_{F,n} \ \&\ \alpha^{\mathbf{B}}(p) \cap B' \neq \emptyset\}.$$

If $\mathbf{B}$ is deterministic and $B' = \{b'\}$, then we can write simply

$$T(\mathbf{B}) = \{p \mid p \in T_{F,n} \ \&\ b' \in \alpha^{\mathbf{B}}(p)\}.$$

Our definitions are rather general because we allow even infinite state sets of automata. But this general case is needed only by some later discussions. In this Part I by automata we always mean finite automata.

Two automata, $\mathbf{A}$ and $\mathbf{B}$ are called *equivalent* if $T(\mathbf{A}) = T(\mathbf{B})$. Two class of automata, C_1 and C_2 are equivalent if for every $\mathbf{A}$ from C_1 there is a $\mathbf{B}$ from C_2, equivalent to $\mathbf{A}$, and conversely. The following statements are well known, cf. [6], [7].

Proposition 1. The classes of deterministic descending, nondeterministic descending and nondeterministic ascending automata are equivalent. Taking an arbitrary automaton from one of these classes, we can effectively construct two automata belonging to the two other classes equivalent to it.

Let $\mathscr{F}_{\mathrm{REC}}$ denote the class of forests, recognizable by deterministic descending automata and $\mathscr{F}_A$ the class, recognizable by deterministic ascending automata.

Proposition 2. $\mathscr{F}_A \subsetneqq \mathscr{F}_{\mathrm{REC}}$.

Since forests are subsets of $T_{F,n}$, we may define the usual set-theoretic operations $\cup$ (union), $\cap$ (intersection) and $^-$ (complementation) on them. Let us now define two more operations the x_i-product and x_i-iteration.

The *x_i-product* of the forests T_1 and T_2, denoted by $T_1 \cdot_{x_i} T_2$, is the forest which can be obtained by replacing every occurence of x_i in some tree from T_1 by a tree from T_2. The *x_i-iteration* of the forest T, denoted by T^{*x_i}, is defined by $T^{*x_i} = \bigcup (T^{k,x_i} | k=0, 1, 2, \dots)$ where

(i) $T^{0,x_i} = x_i$,

(ii) $T^{n,x_i} = T^{n-1,x_i} \cup T \cdot_{x_i} T^{n-1,x_i}$

or $n \in N$. We refer to the operations union, x_i-product and x_i-iteration as *regular operations*. It is well known that $\mathscr{F}_{\text{REC}}$ is closed under the regular operations, but $\mathscr{F}_A$ is not.

Let us take some language $\mathscr{L}$ suitable for describing the sets accepted by automata. In Automata Theory the following two problems play an important role:

(1) Given an automaton A. Describe the set accepted by A in terms of $\mathscr{L}$.

(2) Given a description of the set T in the language $\mathscr{L}$. Construct an automaton accepting T.

The solution is given by the famous Kleene-theorem if $\mathscr{L}$ is the language of regular expressions. Next we review briefly the generalization of this theorem for deterministic descending automata. First of all, we have to define the language $\mathscr{L}$ of (generalized) regular expressions.

Let F be an arbitrary type. The set $\mathscr{R}_F$ of *regular F-expressions* is the smallest set for which

(i) $\emptyset \in \mathscr{R}_F$,

(ii) if $p \in T_F$, then $p \in \mathscr{R}_F$, and

(iii) if $\mathscr{K}_1, \mathscr{K}_2 \in \mathscr{R}_F$ and $i \in N$, then $(\mathscr{K}_1 + \mathscr{K}_2)$, $(\mathscr{K}_1 i \mathscr{K}_2)$, $(\mathscr{K}_1)^{*i} \in \mathscr{R}_F$ hold.

If F is known from the context, we speak about regular expressions simply. An occurrence of the variable x_i in $\mathscr{K}$ is called *bounded* if this occurrence is in $\mathscr{K}_1$ for a subexpression $\mathscr{K}_1 i \mathscr{K}_2$ of $\mathscr{K}$. All other occurrences of x_i are called *free*. x_i is a *free variable* of $\mathscr{K}$ if x_i has at least one free occurrence in $\mathscr{K}$. $\mathscr{K}$ is an *n-regular expression* if all its free variables are in X_n.

Each regular expression $\mathscr{K} \in \mathscr{R}_F$ denotes a forest $|\mathscr{K}|$ given by the following rules:

(i) if $\mathscr{K} = \emptyset$, then $|\mathscr{K}|$ is the empty forest,

(ii) if $\mathscr{K} = p (p \in T_F)$, then $|\mathscr{K}| = \{p\}$,

(iii) if $\mathscr{K} = (\mathscr{K}_1 + \mathscr{K}_2)$, then $|\mathscr{K}| = |\mathscr{K}_1| \cup |\mathscr{K}_2|$,

(iv) if $\mathscr{K} = (\mathscr{K}_1 i \mathscr{K}_2)$, then $|\mathscr{K}| = |\mathscr{K}_1| \cdot_{x_i} |\mathscr{K}_2|$,

(v) if $\mathscr{K} = (\mathscr{K}_1)^{*i}$, then $|\mathscr{K}| = |\mathscr{K}_1|^{*x_i}$.

T is a n *n-regular forest* if $T = |\mathscr{K}|$ for some n-regular expression $\mathscr{K}$. Moreover, T is a *regular forest* if T is n-regular for some $n \in N$. Finally, $\mathscr{F}_{\text{REG}}$ stands for the class of all regular forests.

Proposition 3. (Kleene-theorem, cf. [7]) $\mathscr{F}_{\mathrm{REG}}=\mathscr{F}_{\mathrm{REC}}$. More precisely, for an arbitrary deterministic descending automaton one can effectively construct a regular expression denoting the forest accepted by this automaton, and conversely.

Regular tree grammars are direct generalizations of the well known regular (string) grammars. The following definition is equivalent to the usual one, cf. [2].

Let F be a type, a *regular F-grammar* is a system $\Gamma=\langle Q, F, P, S\rangle$, where

(i) Q is a finite nonempty *set of nonterminal symbols,*

(ii) $S\subseteq Q$ is the *set of initial symbols,*

(iii) P is a finite *set of rewriting rules* of the form

$$q\to x_i\ (q\in Q,\ x_i\in X)\quad \text{or}\quad q\to f(q_1, \dots, q_k)\ (q, q_1, \dots, q_k\in Q,\ f\in F_k).$$

If all variables occuring in the rules of P are from X_n, then we say that Γ is an *n-ary regular F-grammar.* When n and F are not specified, we speak about *regular (tree) grammars.* The n-ary regular F-grammar Γ induces a binary relation $\Rightarrow_\Gamma$ on the set $T_{F,Q\cup X_n}$, $t\Rightarrow_\Gamma r$ iff r can be obtained from t by replacing a nonterminal q in t with the right side of some rule $q\to f(q_1, \dots, q_k)$ or $q\to x_i$ from P. Let $\Rightarrow_\Gamma^*$ denote the reflexive, transitive closure of the relation $\Rightarrow_\Gamma$.

The set $T(\Gamma)=\{p\,|\,p\in T_{F,n}\,\&\,(\exists s)(s\in S\,\&\,s\Rightarrow_\Gamma^* p)\}$ is the *forest generated by* Γ. T is called *generable,* if $T=T(\Gamma)$ for some regular grammar Γ. $\mathscr{F}_{\mathrm{GEN}}$ denotes the class of all generable forests.

Proposition 4. (Brainerd [2]) $\mathscr{F}_{\mathrm{REC}}=\mathscr{F}_{\mathrm{GEN}}$.

3. Closed forests

In this section we show that every forest $T\in\mathscr{F}_A$ contains all trees composed from the paths of some trees belonging to T. We shall use this characteristic feature for deriving a connection between $\mathscr{F}_{\mathrm{REC}}$ and $\mathscr{F}_A$.

With every type F we associate a new type $\delta(F)$ of unary operational symbols in the following way:

(i) if $f\in F_k$, then $\delta(f) = \{f_1, f_2, \dots, f_k\}$,

(ii) if $f\neq g$, then $\delta(f)\cap\delta(g) = \emptyset$,

(iii) $\delta(F) = \bigcup(\delta(f)|f\in F)$.

Now let us define the functions δ_i: $\mathscr{P}(T_F)\to\mathscr{P}(T_{\delta(F)})$ for all $i\in N$ as follows

(1) $\delta_i(x_k) = \begin{cases} x_k & \text{if } i = k,\\ \emptyset & \text{otherwise,}\end{cases}$

(2) $\delta_i(f(p_1, \dots, p_k)) = f_1(\delta_i(p_1))\cup f_2(\delta_i(p_2))\cup\dots\cup f_k(\delta_i(p_k))$,

(3) $\delta_i(T) = \bigcup(\delta_i(t)|t\in T)$.

Let δ: $\mathscr{P}(T_{F,n})\to\mathscr{P}(T_{\delta(F),n})$ be the function for which

$$\delta(T) = \bigcup(\delta_i(T)|i = 1, 2, \dots, n).$$

Speaking informally, $\delta(t)$ consist of all words which can be read along the paths of t. The inserted indices show the position of the node to be visited next. The elements of $\delta(T)$ can be regarded as words of the free semigroup generated by $\bar{\delta}(F)\cup X_n$ as well.

Lemma 1. The function δ is monotone and commutes with the regular operations, i.e.

(i) if $T_1\subseteq T_2$, then $\delta(T_1)\subseteq\delta(T_2)$,

(ii) $\delta(T_1\cup T_2)=\delta(T_1)\cup\delta(T_2)$,

(iii) $\delta(T_1\cdot_{x_i}T_2)=\delta(T_1)\cdot_{x_i}\delta(T_2)$,

(iv) $\delta(T^{*x_i})=(\delta(T))^{*x_i}$.

Proof. (i) and (ii) are obvious. For verifying (iii), first we show the inclusion $\delta(T_1)\cdot_{x_i}\delta(T_2)\subseteq\delta(T_1\cdot_{x_i}T_2)$. If $g\in\delta(T_1)\cdot_{x_i}\delta(T_2)$, then we must distinguish the following two cases:

1) $g\in\delta_j(T_1)$ and $j\neq i$. This directly implies $g\in\delta_j(T_1\cdot_{x_i}T_2)$. Therefore, $g\in\delta(T_1\cdot_{x_i}T_2)$.

2) $g=g_1\cdot_{x_i}g_2$ where $g_1\in\delta_i(t_1)$ for some $t_1\in T_1$ and $g_2\in\delta_j(t_2)$ for some $t_2\in T_2$. Then $g\in\delta_j(t_3)$ holds for the tree $t_3=t_1\cdot_{x_i}t_2\in T_1\cdot_{x_i}T_2$. Thus $g\in\delta(T_1\cdot_{x_i}T_2)$. The inclusion $\delta(T_1\cdot_{x_i}T_2)\subseteq\delta(T_1)\cdot_{x_i}\delta(T_2)$ can be verified in a similar way.

Using (ii), (iii) and the identity $T^{n,x_i}=T^{n-1,x_i}\cup T\cdot_{x_i}T^{n-1,x_i}$ it is easy to prove by induction on n that (iv) holds, too. □

REMARK. From Lemma 1 it follows that the regularity of T implies the regularity of $\delta(T)$. The converse is not true. For this it is enough to consider the forest T of all balanced trees in $T_{F,n}$.

Let $\delta^{-1}\colon\mathcal{P}(T_{\bar{\delta}(F),n})\to\mathcal{P}(T_{F,n})$ denote the inverse of δ, i.e., $\delta^{-1}(U)=\{t\mid\delta(t)\subseteq U\}$ for every $U\subseteq T_{\bar{\delta}(F),n}$. We define the operator $\Delta\colon\mathcal{P}(T_{F,n})\to\mathcal{P}(T_{F,n})$ as the composition $\delta^{-1}\cdot\delta$

$$\Delta(T)=\{t\mid\delta(t)\subseteq\delta(T)\}\quad(T\subseteq T_{F,n}).$$

Lemma 2. Δ is an algebraic closure operator on $\mathcal{P}(T_{F,n})$, that is

(i) $T_1\subseteq T_2\Rightarrow\Delta(T_1)\subseteq\Delta(T_2)$,

(ii) $T\subseteq\Delta(T)$,

(iii) $\Delta(T)=\Delta(\Delta(T))$,

(iv) if $t\in\Delta(T)$, then $t\in\Delta(T_1)$ for some finite $T_1\subseteq T$.

Proof. Obvious. □

We say that $T\subseteq T_{F,n}$ is *Δ-closed* if $\Delta(T)=T$. Let $\mathcal{F}_c$ denote the class of all Δ-closed forests.

Lemma 3. If F has at least one non-unary operational symbol then $\mathcal{F}_c$ and $\mathcal{F}_{\text{REC}}$ are incomparable.

Proof. For the sake of simplicity assume that $F_2 \neq \emptyset$ and $f \in F_2$. Let

$$T_1 = \{f(x_1, f(x_1, x_1)), f(f(x_1, x_1), x_1)\}$$

and

$$T_2 = T_3 \cap T_4, \text{ where}$$

$$T_3 = T(\Gamma), \quad \Gamma = \langle\{S, R\}, \{f\}, \{S \to f(R, S), S \to x_1, R \to x_1\}, S\rangle$$

and

$T_4 = \{p \mid p$ is composed from an arbitrary number of f's and a prime number of x_1's$\}$.

$T_1 \in \mathcal{F}_{\mathrm{REC}}$ is obvious, but $T_1 \notin \mathcal{F}_c$ because of $f(x_1, x_1) \in \Delta(T_1)$ and $f(x_1, x_1) \notin T_1$. On the other hand $T_2 \in \mathcal{F}_c$, but $T_2 \in \mathcal{F}_{\mathrm{REC}}$ would lead to a contradiction. □

Now we generalize the construction of the well-known powerset automaton for ascending tree automata. Let $\mathbf{A} = \langle \mathfrak{A}, A', \mathbf{a}\rangle$ be an n-ary nondeterministic ascending F-automaton. The *powerset automaton,* belonging to $\mathbf{A}$ is the n-ary deterministic ascending F-automaton $\mathbf{PA} = \langle \mathfrak{P}\mathfrak{A}, A', \mathbf{b}\rangle$, where $\mathfrak{P}\mathfrak{A} = \langle \mathcal{P}(A), F\rangle$ is the deterministic ascending F-algebra with operations $f^{\mathfrak{P}\mathfrak{A}}$ defined by

$$f^{\mathfrak{P}\mathfrak{A}}(C) = \prod_{i=1}^{k} \left(\bigcup(\pi_i(f^{\mathfrak{A}}(c)) \mid c \in C)\right)$$

for every $C \subseteq A$ and $f \in F_k$, $\mathbf{b} = (B^{(1)}, B^{(2)}, \ldots, B^{(n)})$ with $B^{(i)} = (D \mid D \subseteq A \;\&\; D \cap A^{(i)} \neq \emptyset)$ and π_i denotes the ith projection.

Now we recall some concepts from [4]. For any state a of the n-ary nondeterministic ascending F-automaton $\mathbf{A}$ we define

$$T(\mathbf{A}, a) = \{p \mid p \in T_{F,n} \;\&\; a \in \alpha^{\mathbf{A}}(p)\}.$$

A state a is called a *0-state* if $T(\mathbf{A}, a) = \emptyset$. We say that $\mathbf{A}$ is *normalized* if, for all $a \in A$, $n \in N$ and $f \in F_n$ either all of the components of $f^{\mathfrak{A}}(a)$ are 0-states or none of them is a 0-state.

Lemma 4. For any nondeterministic ascending automaton $\mathbf{A}$ an equivalent normalized nondeterministic ascending automaton $\mathbf{A}^*$ can be constructed.

Proof. This lemma is a generalization of Theorem 3 in [4] for nondeterministic automata. The proof can be performed similarly. □

Lemma 5. For every normalized nondeterministic ascending automaton $\mathbf{A}$, $T(\mathbf{PA}) = \Delta(T(\mathbf{A}))$.

Proof. We shall verify the inclusion $T(\mathbf{PA}) \subseteq \Delta(T(\mathbf{A}))$ first. Let $t \in T(\mathbf{PA})$ and $g \in \delta_j(t)$ for some $1 \leq j \leq n$. It follows from the definition of $T(\mathbf{PA})$, that in this case we can correspond to the branches of g a sequence $a_0, a_1, \ldots, a_v$ of states from A such that

(i) $a_k \in A$ $(0 \leq k \leq v)$, $a_0 \in A'$ and $a_v \in A^{(j)}$,

(ii) $a_{i+1} \in \pi_k(f^{\mathfrak{A}}(a_i))$ if a_i $(0 \leq i \leq v-1)$ corresponds to the branch labelled by f_k.

We can complete g to a tree $\bar{t}$ accepted by $\mathbf{A}$ because $\mathbf{A}$ is normalized. Thus $g \in \delta_j(\bar{t}) \subseteq \delta_j(T(\mathbf{A}))$ which implies $t \in \Delta(T(\mathbf{A}))$ since g and t were arbitrary.

The reverse inclusion can be verified in a similar manner. □

REMARK. We show by a simple counterexample that the preceding Lemma does not hold for unnormalized automata. Let $\mathbf{A}=\langle\mathfrak{A}, A', \mathbf{a}\rangle$ be the automaton where

$$A=\{a_0, d\},\quad F=F_2=\{f\},\quad n=1,$$
$$f^{\mathfrak{A}}(a_0)=\{(d, a_0), (a_0, d)\},\quad f^{\mathfrak{A}}(d)=\{(d, d)\}$$
$$A'=A^{(1)}=\{a_0\}.$$

Then $f(x_1, x_1)$ is in $T(\mathbf{PA})$, but not in $\Delta(T(\mathbf{A}))$. □

If we apply the construction of **PA** for a *deterministic* automaton **A**, we get an automaton equivalent to **A**. From this assertion, using Lemma 5, it directly follows.

Corollary 6. The regular forest T can be recognized by a deterministic ascending automaton iff T is closed.

Corollary 7. It is effectively decidable for every regular forest T whether T is recognizable by a deterministic ascending automaton.

Proof. Let $T=T(\mathbf{A})$, where **A** is a nondeterministic normalized ascending automaton. In this case, by Corollary 6, $T\in\mathcal{F}_A$ iff $T(\mathbf{A})=T(\mathbf{PA})$. By Proposition 2.1 we can construct two deterministic descending automata equivalent to **A** and **PA**, respectively. For this type of automata the equivalence problem is decidable. □

4. D-regular operations and the generalized Kleene-theorem

It can easily be seen that $\mathcal{F}_A$ is not closed under the regular operations. In fact, it is not closed under the polinomials of these operations, either. More precisely, this is true for almost all polinomials but those unary ones constructed by unions only. Since $\mathcal{F}_A$ can be obtained as the Δ-closure of $\mathcal{F}_{\text{REG}}$ it seems reasonable to define 'D-regular operations' by combining Δ and the regular operations. This enables us to derive a 'Kleene theorem' for $\mathcal{F}_A$.

Now let us define the *D-regular operations* for any T_1 and T_2 from T_F as follows:

(1) Δ-union $\widehat{\cup}$: $T_1\widehat{\cup}T_2=\Delta(T_1\cup T_2)$,

(2) (Δ, x_i)-product $\odot_{x_i}$: $T_1\odot_{x_i}T_2=\Delta(T_1\cdot_{x_i}T_2)$,

(3) (Δ, x_i)-iteration $\widehat{*}_{x_i}$: $(T)^{\widehat{*}x_i}=\Delta(T^{*x_i})$,

where on the right sides $\cup$, $\cdot_{x_i}$ and $*_{x_i}$ stand for the ordinary regular operations.

Lemma 1. The D-regular operations preserve regularity and recognizability by deterministic ascending automata.

Proof. Follows from Proposition 2.3 and Corollary 3.6. □

Lemma 2. For D-regular operations the following identities hold

(i) $\quad T_1\widehat{\cup}T_2=\Delta(T_1)\widehat{\cup}\Delta(T_2)$,

(ii) $\quad T_1\odot_{x_i}T_2=\Delta(T_1)\odot_{x_i}\Delta(T_2)$,

(iii) $\quad (T)^{\widehat{*}x_i}=\big(\Delta(T)\big)^{\widehat{*}x_i}$.

Proof. These identities can be derived applying the function δ^{-1} for both sides of the equations

$$(*) \qquad \delta(T_1 \cup T_2) = \delta(\Delta(T_1) \cup \Delta(T_2)),$$

$$(**) \qquad \delta(T_1 \cdot_{x_i} T_2) = \delta(\Delta(T_1) \cdot_{x_i} \Delta(T_2)),$$

$$(***) \qquad \delta((T)^{*x_i}) = \delta(\Delta(T)^{*x_i}).$$

We know from Lemma 3.1 that δ commutes with the regular operations. It is also easily seen that $\delta(\Delta(T)) = \delta(T)$ holds for every forest T. These assertions imply the identities $(*)$—$(***)$. $\square$

Next we introduce a new *'D-regular interpretation'* $\|\mathscr{K}\|$ of the regular expression $\mathscr{K}$

(i) if $\mathscr{K} = \emptyset$, then $\|\mathscr{K}\|$ is the empty forest,

(ii) if $\mathscr{K} = p (p \in T_F)$, then $\|\mathscr{K}\| = \{p\}$,

(iii) if $\mathscr{K} = (\mathscr{K}_1 + \mathscr{K}_2)$, then $\|\mathscr{K}\| = \|\mathscr{K}_1\| \widehat{\cup} \|\mathscr{K}_2\|$,

(iv) if $\mathscr{K} = (\mathscr{K}_1 i \mathscr{K}_2)$, then $\|\mathscr{K}\| = \|\mathscr{K}_1\| \odot_{x_i} \|\mathscr{K}_2\|$,

(v) if $\mathscr{K} = (\mathscr{K}_1)^{*i}$, then $\|\mathscr{K}\| = \|\mathscr{K}_1\|^{\widehat{*} x_i}$.

T is a *D-regular forest* if $T = \|\mathscr{K}\|$ for some regular expression $\mathscr{K}$.

Lemma 3. For every regular expression $\mathscr{K}$, $\|\mathscr{K}\| = \Delta(|\mathscr{K}|)$.

Proof. By induction on the number of the symbols of regular operations in $\mathscr{K}$ using Lemma 2. $\square$

Theorem 4. The forest T is recognizable by deterministic ascending automata iff T is D-regular, and this connection is effective.

Proof. (1) Let T be given by the deterministic ascending automaton **A**. According to Proposition 2.1 and 2.3 we can effectively construct a deterministic descending automaton **B** and a regular expression $\mathscr{K}$ such that

$$T = T(\mathbf{A}) = T(\mathbf{B}) = |\mathscr{K}|.$$

T is closed (see Corollary 3.6) thus $T = \Delta(T) = \Delta(|\mathscr{K}|)$. But this yields, by Lemma 3, $T = \|\mathscr{K}\|$.

(2) Now let us assume that $T = \|\mathscr{K}\|$ for the regular expression $\mathscr{K}$. By Proposition 2.1 and 2.3 we can effectively construct a nondeterministic ascending automaton **C** accepting $|\mathscr{K}|$. **C** can be assumed to be normalized (see Lemma 3.4). Proceeding as in Lemma 3.5 we get the deterministic ascending powerset automaton **PC** for which $T(\mathbf{PC}) = \Delta(T(\mathbf{C})) = \Delta(|\mathscr{K}|) = \|\mathscr{K}\|$ holds. $\square$

REMARK. In the preceding proof we used Proposition 2.3 in both directions. Theorem 4 could be proved without it, but in that case the proof would be more lengthy and difficult.

5. D-regular tree grammars

In Proposition 2.4 we have established a close connection between forests generable by regular tree grammars and forests recognizable by deterministic descending automata. In this section we shall give a similar characterization for forests accepted by deterministic ascending automata. To this we define a special kind of regular tree grammars.

The regular tree grammar $\Gamma=\langle Q, F, P, S\rangle$ is called *deterministic regular*, or briefly *D-regular* if

(i) S is a singleton and

(ii) for every nonterminal q and operational symbol f there is exactly one derivation rule in P, whose left side is q and whose right side begins with f.

Theorem 1. The forest T is recognizable by deterministic ascending automata iff T is generable by D-regular tree grammars.

Proof. The well-known constructions of converting a regular tree grammar into an equivalent (nondeterministic) ascending automaton and vice versa can be used. The only thing to be noted is that the assumptions (i) and (ii) in the definition guarantee the preservation of the determinism in both directions. □

Corollary 2. For every regular tree grammar Γ one can decide effectively whether $T(\Gamma)$ can be generated by D-regular tree grammars.

Proof. Since a nondeterministic ascending automaton accepting $T(\Gamma)$ can effectively be constructed (cf. Proposition 2.1 and [2]) Corollary 3.6 and Theorem 1 immediately imply our Corollary. □

Acknowledgement. The author wishes to thank Professor M. Steinby for his valuable suggestions especially for those concerning Theorem 4.4 and for providing the example for the Remark in Section 3.

DEPT. OF COMPUTER SCIENCE
A. JÓZSEF UNIVERSITY
ARADI VÉRTANUK TERE 1.
SZEGED, HUNGARY
H—6720

References

[1] Brainerd, W. S., The minimization of tree-automata, *Inform. and Control*, v. 13, 1968, pp. 484—491.
[2] Brainerd, W. S., Tree generating regular systems, *Inform. and Control*, v. 14, 1969, pp. 217—231.
[3] Doner, J., Tree acceptors and some of their applications, *J. Comput. System. Sci.*, v. 4, 1970, pp. 406—451.
[4] Gécseg, F. and M. Steinby, Minimal ascending tree automata, *Acta Cybernet.*, v. 4, 1978, pp. 37—44.
[5] Grätzer, G., *Universal algebra*, Van Nostrand, Princeton, N. J., 1968.
[6] Magidor, M. and G. Moran, Finite automata over finite trees, *Tech. Rep. Hebrew Univ.*, Jerusalem, No. 30, 1969.
[7] Thatcher, J. W. and J. B. Wright, Generalized finite automata theory with an application to a decision problem of second order logic, *Math. Systems Theory*, v. 2, 1968, pp. 57—81.

(Received Oct. 24, 1979)